An Introduction to Content Management

Single-Source Document Design Criteria and Workflow

By
Stephen J. Fishman

©Copyright 2001 Stephen J. Fishman

Table of Contents

Overview

Introduction

The development of XML has been driven by three factors - the increasing demands by users for documents that can present information in more interesting ways, the escalating need to share common information among many authoring communities and the ever-growing spectrum of display devices.

Taking Advantage of Technology

On the whole, users are more savvy than in the past as regards documentation and the many forms it can take. User demands are high for feature-rich documents that can, for excample, augment assembly instructions with animation to help them visualize the process.

For many years, computers and word processing software have provided an easy method of sharing information with others via floppy disk, removable media and email, among others. A predictable extension of this situation is the desire of documentation managers to share information across departmental boundaries within corporations with an eye to reducing the time.and budget needed to deliver related documents containing the same information.

Finally, the proliferation of more specialized and compact computers has spawned the desire for information that can be viewed on a variety of display devices from a single source.

Taken together, the key issue that emerges from current trends is sharing common data and, to a large extent, XML addresses this issue better than any other markup language currently being used.

XML Tags

XML tags are similar to HTML tags in that attributes are placed between <> symbols. The notation convention is the same, with tags used before and after a component, such as <topic></topic> or <step-list></step-list>.

As mentioned before, most XML tags are defined by the user and identify types of data or the meaning of the content.

Metadata

Metadata is simply data about data. Metadata is the non-display and non-printing information that provides the particulars about an XML file. Metadata typically contains such fields as the author's name, the department, the most recent edit date, the original date of creation and so forth.

Metadata is often used as search criteria when locating a module about on a specific topic or procedure.

XML Defined

XML is not a language but rather a syntax for specifying a language that is adaptable and customizable by the author. XML – eXtensible Markup Language – is a subset of HTML and an outgrowth of SGML. XML is similar to HTML in that tags are used to identify various components within the document, but that is where nearly all the comparison ends.

HTML tags identify the type of data – a paragraph, a graphic, a table but not the information type or content. On the other hand, XML tags identify the type and meaning of the content - procedure, part number, title, label, etc.

In addition, XML tags can be created by the user to identify the content relative to a specific industry or business. For example, XML tags can be created to isolate and reuse aseembly procedures, maintenance schedules, theory of operation, dates, etc. Because of this, searches can be made on a particular content and information type that exludes superfluous data and elimnates most of the problems associated with searches that locate too much data.

Methodologies

XML authoring can be as basic as opening an editor and using the existing tags and stylesheets, or it can be incredibly complex. The potential complexity lies in the number of layers of control and the dregree to which the system is customized.

In the XML environment, it is not so much the tools that create the challenges as the methodologies, and the pitfalls and problems increase exponentially as the complexity increases.

Imagine an inverted pyramid. As each layer is added, both the cost and the complexity spiral upward exponentially. These layers include:

- XML authoring
- XML conversion from Word, especially tables
- Granularity/modularity standards
- Creating new tags
- FOSIs and style sheets
- DTDs
- Multiple format publishing
- Access and version control

Each of these layers introduces its own benefits and challenges relative to access rights, editing and approval security, cross-project linking, etc. These layers of methodology drive the capability of the system but, simultaneously, create deadline and budget obstacles.

Information Products

As the need for increased communication efficiency has grown, the demand for timely and function-specific documentation has also grown. Although still widely used, traditional user manuals and product brochures are increaingly giving way to more sophisticated formats of documentation and information transfer.

Many of these documents are delivered by means of a web browser and can include static text, animated images, movie clips and user-entered data. Some types of browser-delivered documents include:

- Manuals – user guides and installation instructions
- Courseware
- White papers and reports
- Engineering and manufacturing specs
- Marketing documents
- IETMs (interactive electronic technical manuals)

Of these various types, it is the IETM that presents the greatest opportunity for accurate, easily comprehended information presentation while, at the same time, introducing the greatest challengesd and obstacles to its production.

Single-Source Documentation

Reusability

The idea of single source documentation is not new. As far back as I can remember, single source information was a part of my life, and I'm sure it was a part of yours, too. To prove this to yourself you only need to remember those instances in your childhood when your parents said something like, "How many times do I have to tell you?" or "If I've told you once I've told you a thousand times,…!" In a fundamental way, this is information reuse.

Or how about television commercials? Often, a product will be advertised in several different ways in an attempt to reach the widest possible audience. Each commercial contains the same verbiage but mixed with different images or, conversely, the pictures and film clips are rearranged to support copy that is focused on a particular segment of the viewing audience. In both instances, single source documentation is applied.

The most common example of this approach are ads for automobiles. In one campaign from the 1970s, a shady character by the name of Joe Isuzu oversells the merits of this maker's product while, simultaneously, text at the bottom of the screen tells the audience that he is lying. This series of commercials for Isuzu automobiles was one of the most successful campaigns of that decade.

There are two fundamental reasons for employing this philosophy of information reuse, namely cost and efficiency. Costs can be reduced by not having to rewrite copy and recreating the images while, simultaneously, reducing the time needed to produce different versions of the same information when existing material can be assembled in multiple ways.

That said, not all information presentation lend itself to reuse. Text and graphics produced for an engineering audience may have little application for marketing, and classroom instruction presents more basic information than what might be done on site in a hands-on environment. Even so, the vast majority of professionally authored information should normally be expected to have more than one use.

What "Multiple-Use" Means

When we speak of reusable documentation, we're usually referring to one of two issues, either assembling information in multiple ways or publishing information in multiple ways.

Document Assembly

In this context, reusability applies to the usefulness of the information relative to more than one audience or purpose. There are many examples of this that may be worth considering, such as:

- A user guide that can be broken down into lessons for classroom presentation
- An engineering design specification document that can be used, in part, by marketing to help sell the product or service.
- Bills-of-material that form the basis of a maintenance guide
- Web-delivered instruction that Human Resources can use for personnel training records
- A revision of an existing document that needs only a partial update to be brought current.

Document Publishing

Without exception, documents are written for a specific audience and method of presentation. Whether the material will be viewed online or as hardcopy, integrated into software or provided as standalone help, information presentation is a critical aspect of document design. Currently, there are three primary presentation methods, and each has a file format that is unique:

- Hardcopy – Acrobat PDF or MS Word
- Web-delivered – HTML or XML
- Live presentation – MS PowerPoint

Of these, the most widely used format is PDF due to its "sealed" un-editable form. But PDF is typically not the best choice when delivering information across the Internet, especially when considering the newest innovations in document design that can include animations and video.

Single-source documentation offer the capability to publish a wide range of formats uniquely tailored to the audience and the method of delivery.

IETM

What it is

This one acronym, more than any other, symbolizes the benefits of single-source authoring and multiple-format publishing. When spoken, the acronym sounds quite humorous – “I eat ‘um!” When fully written, the acronym is “Interactive Electronic Technical Manual.” This concept of information design has the potential to deliver the most informative, interesting and entertaining documents that the in Internet can host.

The IETM concept is intended exclusively for electronic delivery, either across the web or on a CD. These documents can be quite large, which means the author must decide at the design stage how the information will be delivered. For example, delivery on a CD usually requires that the user must install - or at a minimum download – the contents to the local hard drive.

On the other hand, efficient delivery across the Internet demands small file sizes and limited file formats in order to minimize load times and accelerate the information transfer process.

The greatest strength of IETMs is its ability to mix video, sound, animation, static images and text as a means of tailoring the information to the method of presentation. For example, an animated product assembly/disassembly procedure is far more self-explanatory and more quickly understood than the traditional exploded view presented as a line drawing.

Traditional “flat” hardcopy documents have existed for years, with improvements in visual clarity the primary improvement over their predecessors. An IETM creates an active, if not interactive, environment that more closely approximates what most experts agree is the most effective way to present ideas and information – visual action. As a matter of design, IETMs offer a visual variety to the learner that helps to maintain a high level of interest regardless of the intrinsic value of the information.

Although the concept of an IETM can be delivered without applying the methodology of single-source documentation, its greatest appeal for large corporations is the IETMs ability to combine interesting presentation with audience-specific information. Certainly, an IETM can be authored with information that is never reused, but that would be an incredible potential waste if portions of the information contained within an ITEM could be reused in other IETMs or traditional documents.

What it isn’t

An IETM can be too much in many information presentations. Recognizing that the acronym includes a letter representing the word “manual,” it is important to recognize that small presentations, single-topic presentations, job aids, quick reference guides and help systems are all examples of misplaced uses of the IETM concept.

In an environment of condensed, short-lived information presentation the IETM format is often not a cost-justifiable approach. Even applying the principles of single-source information architecture, an IETM can be too expensive to produce or too lengthy in production time to meet short-term small budget goals.

Even so, the underlying concept of an IETM can be summed up in one word – variety. Regardless of the budget, making a conscious choice to avoid repetitive methods of presentation can go a long way towards getting the information across to the audience in a way that improves the likelihood of retention.

Layers of Methodology

As mentioned before, the XML authoring environment can be quite complex, and the pitfalls and obstacles increase significantly as the complexity grows. Some typical issues to be addressed include, but are certainly not limited to, the following.

Each of these layers, in turn, introduces its own challenges relative to access rights, approval, cross-project linking, etc.

XML Authoring

Testing the available tools

This poses more problems than you might at first imagine. One of the greatest challenges in any XML strategy is matching vendors that can work together to develop a solution. Often, the perfect authoring tool will not interface with an EDMS, leaving version control and information management unaddressed.

Another major consideration when testing the tools is the matter of flexibility. If an authoring tool cannot be readily customized, there is little promise that it will satisfy the special needs of a changing business climate and upgrades client expectations.

The best example of this type of problem is the issue of tags. A fully-functional authoring tool must provide the means by which new tags can be created as new processes, techmologies and bisness models are developed.

Training the authors

Training is always expensive and time-consuming, and this is especially true of XML authoring. It is undeniable that we are all creatures of habit and, to that end, most of us have a tendency to try and preserve existing ways of doing things. Implementing an XML strategy as a basis for enterprise-wide documentation standards is an exercise in working against this ingrained aspect of human nature.

Content is most often provided by SMEs - developers and engineers - even though writing is not their primary (or even secondary) focus. Although vastly better than in years past, technical SMEs rarely write more than necessary and adding XML authoring into the mix presents the very real possibility that time-sensitive content deliverables will suffer.

To the question of who will train new authors, the answer can only reasonably be trained, seasoned documentation specialists. Engineers may understand the authoring system and developers may feel comfortable with the tag schema, but only experienced documentation people are typicall able to fully integrate the tools and the methodologies. The greatest benefit that a company can generate for its long-term documentation health will be derived from a concerted effort to hire and adequately train its writers, Document Managers, or both.

Hardware requirements

Most of the XML authoring tools currently available will only work correctly on high-end workstations, and some authoring tools may not work at all except on the fastest of PCs. Workstations that were placed into service primarily for using Microsoft Office may be woefully short of the horsepower necessary to run an authoring tool or serve an EDMS.

XML Conversions

In-house or contract?

Document conversions to XML, whether from MS Word or HTML, are tedious, time-consuming and expensive. For these reasons, it is usually more cost-efficient to out-source this work to a vendor that specializes in conversions than to tie up valuable staff resources. This alone, however, will not solve the problem, since the the issues of content accuracy and granularity can only be addressed by SMEs and Document Managers.

Depending upon the industry and the type of information, it may be a good idea to consider staffing up to convert existing documents in-house, thereby keeping sensitive competitive data or design specifications away from contractors, non-disclosure agreements notwithstanding.

How will special cases be dealt with, especially tables?

At present, no reliable method has been developed thast will readily convert MS Word tables to XML. Similarly, only moderate success has been achieved in writing routines for converting Excel tables to XML. This leaves two options, neither of which is completely acceptable, but for different reasons:

1. Capture a table and save it as a graphic image. This image could then imported into the EDMS as an information object that can be linked to a data module.
2. This choice might work for tables containing information that is rarely updated or changed, but it unlikely to satidfy most applications for which tables are used.
3. Create a new table in a data module and rebuild the table one cell at a time. For small tables this may be a good option. For large tables, howver, it is an unacceptable use of time.

What standards will be followed?

A conversion vendor will necessarily have to use the same DTDs and tools that you use in order to assure compatability with your documents. In order to accomplish this, additional temporary licenses may be required for the software tools and additional task-specific workstations may be needed.

Who will train the vendor's authors?

This question is similar to issue of training the company's staff, but with a different consideration. Although it is not common, it is also not unusual for a conversion vendor to shoulder at least a portion of the training cost with its client. This is due in part to the fact that the vendor will fulfill the contract and retain a new skill set it can market to other clients.

Although the client may have to foot the bill for the lion's share of the software tools and training, it is customary for the contractor to supply the necessary hardware. The vendor may be required to invest somewhat large sums for this equipment but, like the training, it can be offered at a later date to other clients. An arrangement such as this should be clearly spelled out during the early negotiating phases since it may have a direct effect on whether or not a vendor is willing to accept a contract.

Information Granularity

The primary goal of standardization is to eliminate, to the degree possible, as many subjective decisions as possible. And no where within the XML documentation environment is there more opportunity for subjective, non-standard writing than when discussing the topic of granularity. Setting policy regarding this issue is fundamental to the development of consistent information modularity.

What is the smallest acceptable information unit?

Although the answer to this question is likely to depend to some degree on the content, generally speaking, the smallest information unit (data madule) is a single complete thought. The problem revolves around how "complete thought" is interpreted.

To some, a complete thought is akin to a content-unit which can contain text and a graphic, text and a table, or only text. If the complete thought includes only text, is a single sentence acceptable as a data module or is this too small?

Although this might appear to be a trite issue on the surface, a policy decision on this question will affect virtually all other decisions regarding granularity, including the review and approval process.

Criteria

Closely tied to the decision about the degree of granularity is the issue of what criteria will be used to determine it. If content is the only criteria, no further discussion on this point is necessary. However, other criteria that might be considered are the type of information, how often the data will be updated and how many resources will be involved in the update process.

For example, if it is anticipated that the data will updated on a quarterly basis, it might be well to consider the authoring this information in ‘isolation’, separated from other related data.

If it is likely that the cooperation of several SMEs will be needed in order to accomplish the update, the information will be most efficiently authored by segregating the data into modules that can be reviewed by as few SMEs as possible in any effort to facilitate getting the changes into the hands of users.

Creating New tags

Who decides?

This question is at the heart of virtually all decisions about the authoring and delivery environment. Ideally, a core of experienced authors and documentation managers would convene to discuss the various needs of users. To find widely disparate needs from one department to the next is the norm, and attempts to find common ground will most often be met with frustration.

Identifying documentation needs either by product or product usage is a necessary first step but, in the end, it is the document management community that must set the standards for tags that must be used by everyone.

What tags are needed?

Once again, this is derived from the product or service and how it is used. For example, if the product is a tool comprised of many components that must be maintained in the field, it would be advantageous to create tags that identify major components and subassemblies.

Another example is a software product that tracks the maintenance schedule of equipment in the field. Tags that can readily identify equipment types, dates and geographic locations builds value into the product as a result of search accuracy and speed.

Attributes and rules

Virtually all XML tags are built with attributes that can govern the information content or type, the way in which the content can be searched and the way in which the data is displayed. Taken together with their attributes and rules, tags can provide a significant layer of standardization across a widely dispersed and diverse organization.

Tag rules can improve document automation and accelerate the authoring process by controlling numbering, default tag nesting and by defining what tags can be inserted into other tags.

For example, a <Title> tag might always include a <Text> tag by default since a title will always contain text. Or perhaps a list identified with an <Ordered list> tag might always be lettered while a <Step list> tag might always produce a sequence that is numbered.

Rules can control the content of a tag by limiting the options of tags that are nested within it. For example, a tag named <Topic> might only offer three options of nested tags, including <Title>, <Text> and <Content unit>. But a <Content unit> tag might allow a dozen or more tags options, including <Title>, <Text>, <Image>, <List>, <Module link>, etc.

Rules can also enhance and expedite the learning curve of new authors by providing a reduced st of options for any given task.

FOSIs (Style Sheets)

Overview

Since XML data contains no formatting information, style sheets are used to format the data to produce electronic, printed and other output. There can be as many style sheets as needed; i.e.: pocketsize and full-size publications, HTML, PDF, etc.

SCREEN stylesheet
PRINT stylesheet

It is important to remember that, usually, none of the FOSI-controlled display types produce the same appearance in the final information product. For this reason, it is a fairly straightforward process to isolate which FOSI is at fault if a display problem occurs.

For example, users might find that the FOSI used in the authoring environment displays numbered lists correctly while, in the PDF output, the numbers are missing from the same lists. Or perhaps graphics are correectly scaled in the HTML output but are oversize in the hardcopy.

This complete separation of one style sheet from another presents both an opportunity and a management challenge. On the upside, a FOSI can be designed to deliver exactly what is needed in any given user application. On the downside, each FOSI must be individually built and edited, typically borrowing little from its bretheren.

Designing and building style sheets

Like the process of developing which tags are needed, the issue of styles should be decided by document managers after input from the documentation community. In this instance, however, experienced authors and should be joined by representatives from marketing who can provide feedback regarding approved corporate fonts, logos, layout and other formatting properties.

In general, a FOSI will have a direct relationship to the intended purpose, the output format and the delivery mechanism. For example, if the intended purpose is to serve users in remote areas, the best output would be a format that delivers excellent hardcopy and, for that, PDF format might be the best choice.

In anther environment, it might be more convenient to deliver the document via the Internet, suggesting an HTML output for users in urban areas who have access to good connectivity.

In both instances, FOSIs provide identical content but in vastly different ways, providing a way for the document to supply maximum benefit to the intended audience.

Integrating FOSIs

Style sheets in XML are commonly referred to as Formatting Output Specification Instances, or (FOSIs). FOSIs control the way in which a document is displayed in the authoring envirnoment as well as how it looks when published. With rare exception, no two instances are alike.

In addition, a FOSI must be integrated into the editing tool so that authors can view their work and get a sense of how the final output might appear. Invariably, the published output is more richly finished with headers and footers, page numbering, icons for warnings and notes, and bookmarks. If there is more than one output available for publishing, there must be a different FOSI for each one.

Often, FOSIs are constructed by vendors other than the suppliers of the authoring and publishing tools. For this reason, it is imperative to select vendors that can work together to achieve a seamlessly integrated whole.

Output formats

As previously mentioned, the best output formats are the ones that deliver the greatest value to the user. Most often, this means the output will be published in two formats – PDF for the Adobe Acrobat Reader® and HTML for a web browser. Although a third ‘format’ is used in the authoring environment it is not a published output and, therefore, not a part of this dicussion.

The choice of which format to use when publishing has as much to do with the desired presentation as with the intended use. For example, if an animated graphic is determined to be the best visual aid for a discussion of equipment assembly or disassembly, then HTML might be the best option. On the other hand, if the most important aspect of a document is its ability to be printed, then without doubt PDF is the format of choice.

Heirarchies

No discussion of document creation and presentation is complete without some consideration for the way in which logical sections are identified and special types of information are set apart.

Depending upon the industry, a document might be segmented into chapters, sections or volumes, and each of these perspectives brings with it considerations about how each division will be identified. The three most common ways in which document segments are idetified are fairly well knowen:

Arabic numbers using a decimal system - Section 1, Sub-section 1.1, Sub-section 1.2.1, etc.

Roman numerals using a scheme of upper case letters and arabic numbers - Section I, Sub-section I.1, Sub-section I.1.1., etc

Roman numerals using a scheme of upper case and lower case letters - Section I, Sub-section I.a, Sub-section I.a.b, etc

No matter the schema, it must be consistent across all FOSIs and implemented in such a way that authors and reviewers cannot alter the system.

Lists

As in traditional authoring, there are potentially three types of lists in an XML document – unordered lists, ordered lists and step lists – and each has its use. It is important to remember that rules can be established within the authoring environment to control whether or not a specific list can be placed within a given tag or data module. This bears directly on the document deconstruction process since it can directly affect where a list must be authored and may present a limitation regarding what type it can be.

An extension of this control is the way in which FOSIs can be designed to display these lists. For example, an unordered list is typically displayed with bullets that are black dots, but they could just as easily be arrows or asterisks. Further, a heirarchal nested list might display bullets for the top level and dashes for the next level down, like this:

- Top level list item
- List item
- List item
  - Second level item
  - List item
  - List item

Similarly, an ordered list can be displayed differently than a step list, with a step list carrying titles for the step number and the action to be taken, as shown below:

Step	**Action to be taken**
1.	User action or task to be accomplished
2.	User action or task to be accomplished
3.	User action or task to be accomplished

Document Type Definitions (DTDs)

What is it?

A Document Type Definition (DTD) is the skeleton upon which an XML document is built, and contains the rules that govern what type of information can be included and where it must be inserted. When authoring in the XML environment, a DTD is essential for both defining the documents and establishing standards within the documentatio community.

A DTD defines the fundamental aspects of the document, including:

- The order and identification of the major sections
- The type of information that each section will contain and the sequence in which the data must appear
- Which information is required and which is optional
- Exclusions and disallowed information

It is the DTD that differentiates the structure and content of a various documents. This is readily apparent by comparing, for example, the DTD for a software manual with that of a hardware manual. There will almost certainly be similar information and perhaps even some text that is identical, but the order of the data, safety issues and how the information is used will often be radically different.

Who decides?

Like FOSIs, one of the first to querstions to ask is who will be involved in developing the structure and rules of the various DTDs that will be needed for the type of documentation the company creates?

Input from authors is key, of course, but perhaps the most influential input should come from the users or implementers of each product or service. The ultimate decisions reagrding the structure of DTDs must come from management but understanding how documents are utilized on a daily basis can lead to a more accurate assessment of user needs and, by extension, drive the authoring process.

Examples of authoring-for-purpose is obvious in the differing approaches demanded of hardware and software products, but DTDs must be more task-specific it they are to be of maximum value to the user. Some types of DTDs include, but are not limited to:

- Software that calibrates and operates tools and machinery
- Software that does not drive external tools
- Hardware operations
- Hardware maintenance
- Online training (OLT) introductory material
- OLT for operationing and using equipment
- OLT for equipment maintenance

What the rules can do

First and foremost in the decision-making process is the issue of standardization. Unlike the saying, ‘He who has the gold makes the rules’, rules for DTDs are primarily driven by considerations of expediency, simplicity and consistency.

Expediency in this instance means that that the author has fewer decesions to make and, as a result, can produce documents in less time.

Simplicity, closely related to expediency, reduces the number of decisions that an author must make relative to the structure and assembly of documents. With built-in rules controlling such things as prohibited links, disallowed tags and limited attributes, authors can more readily focus on the content and its relationship to the whole. A side benefit of a rules-based simplicity is that fewer ways are available for authors to devise ‘work arounds’ when they are tempted to redifine the system.

And, finally, consistency brings with it a predictable result that provides a common look and feel across even the most diverse documentation community. If the rules built into a DTD do nothing else, they control the form and function of the input so that an output FOSI can correctly map from the document’s tags and produce a standardized deliverable.

Constructing the rules

Once decisions have been made regarding what authoring rules will be included in the DTDs, the next step is to actually integrate the rules sets into each DTD. Invariably, this is accomplished by either the vendor of the authoring tool or a vendor specializing in the constructio of DTDs for XML documentation. In both cases, the DTDs and their rule sets must seamlessly integrate with the authoring tool.

This requirement that the XML editor and the DTDs merge without problems is one of the driving forces behind selecting both the vendor of the editor and the vendor of the DTDs. Given the fact that complex XML authoring is a relatively new technology and methodology, it is not surprising that not all vendors’ products will work with other products.

For example, the combination of the SigmaLink EDMS and the Arbortext Epic XML Editor are often chosen because of the willingness of the two vendors to work together to create a convenient, seamless user interface that relies on programming and design than on user adaptability and knowledge. The result of such a collaboration is another layer of standardization and simplicity.

Publishing

Output formats

As previously mentioned, the most common output formats are PDF and HTML, but this only speaks to the overall result. In order to achieve this, however, additional issues must be considered first and chief among them is the issue of graphics.

Graphics can be used in most any format and the output FOSI will correctly deliver documents containing them but only if the graphics have first been correctly prepared.

Graphics formats

Every DTD has rules about what graphics formats can be used and these rules are always supported by the EDMS. In most documentation environments – XML or traditional - the common formats of choice are JPG and GIF but these may not represent the best choices for XML.

Currently, there is an initiative afoot by Compuserve® to collect fees for the use of the GIF graphics format on the Internet, since this format is the result of a proprietary algorithm developed by Compuserve decades ago. And, although this is an excellent format for graphics delivered for use within a web browser, it is by no means the only format available nor is it perhaps the best one.

New graphics formats have emerged that are open formats for which no fee is due and, among them, the most popular formats are PNG and SVG. These two formats deliver high resolution scalable images with the same, or nearly the same, number of colors as GIF images.

Currently, the Epic editor (among others) can correctly output PNG images in a FOSI but cannot correctly display this format while authoring the document. Other formats for graphics, such as PDF, can also be used provided the DTD permits the use of the format and the EDMS is configured to track images for the Acrobat Reader.

Access Rights and Version Control

Version control

Version control has been used for many years to manage the development of software and protect access to the program components of an application while it migrates through its various stages from inception to delivery. The same is true when the principle is applied to the components that comprise a document prior to publishing.

In the XML environment, a document is built from modules that are authored as components of the finished product, each of whicj must pass through numerous revisions and approvals. These modules must be tracked as they progress from one version to another and one level to the next. It is not unusual for a single module to have as many as ten, or even fifteen, revsions before being promoted to the next level.

In addition, even after the approval process has been completed, version control is the mechanism that assures the document will be assembled from only the most current - and officially approved – modules.

Which vault system?

The importance of a vault system that can be easily customized is not to be underestimated. Without this inherent flexibility, it is impossible to implement a workable solution that integrates the authoring environment with reliable version control of document components.

Database maintenance

Access rights

This is discussed in detail in the section about User Groups, and controlled by a member of the Admin group, a Document Manager, or both.

Editing access

This is discussed in detail in the section titled Roles and Responsibilites.

User types

Reader – Read-only access, typically users in the field

Author – Document creator

Project Manager – Document validator

Publisher – Document assembly and delivery

Documentation Manager - Overall control of content and delivery schedules

Database Administrator – Maintains the database (the vault)

For more information, see the section titled User Groups.

Current Issues

The Problems

Speed, or lack thereof

Currently, some of the most costly aspects of information product delivery are delays at the the point of commercialization and/or distribution. Often, the document delivery mechanism is accepted as a given either due to inertia or because of a lack of perceived need.

All efforts to provide comprehensive and accurate documention are obviated if the informatin doesn’t reach the user on a timerly basis. This would appear to be an obvious point, but distribution often remains unaddressed until a major system overhaul is considered.

Maintainance difficulties

Technical documents especially can suffer from this malady. Often, many SMEs must work in concert before an effective document, or suite of documents, is ready for release. Motivating and organizing a group of engineers or developers is typically the responsibility of management because of the degree of focus it can draw away from other priorities.

In addition, data may have to be field tested or otherwise verified before it can be included in a document update, extending the disparity between content creation and delivery to users.

Excessive turnaround time

Even if data and new information is readily available, the resources needed to turn the data into a useful information product may not be readily available. This can be the result of personnel turnover, the implementatin of new tools and existing priorities.

Due to no fault of their own, documentation specialists can often find themselves in the position of having to satisfy the simultaneous demands of several diffferent entities. Once again, it is management that must distate the priorities.

Inefficient distribution

As mentioned before, distribution is an often overlooked part of the total problem. If a single distribution system is in place, bottlenecks may be encountered in uploading documents to a central location, producing hardcopy or burning CDs. This can be exacerbated by order processing time, shipping delays and approval processes.

Cost

Duplication of effort

The lack of documentation standards within an organization, especially a large corporation, carries the greatest potential for loss.

Duplication of content authoring, reformatting for multiple applications and small,isolated puchases all contribute to escalating production costs.

Systems not automated

Although standards must be in place before the advantages of automation can be fully realized, the lack of automated publishing and delivery systems represents the next largest potential for exceeding a documentation budget.

Manual generation of multiple output formats

Although related to the lack of automation, this speaks directly to the contention that, given the chance, each business entity would have its own unique information products.

The establishment of standards certainly works to limit the number of special formats and automation reduces the cost of delivering such products, but controlling the number and type of permutations is the key to reducing costs in a multiple-segment envirnment.

Limited data re-use

It makes sense that the cost of any product decreases as the number of uses increases. Yet, until recently, this was never applied to document content. But if information can be authored in such a way that re-use is both practical and easy, the cost of delivering that information is reduced with each successive use.

Quality

Process standards

The lack of process standards is one of the biggest obstacles to delivering information products of consistenly high quality. From what size the hardcopy will be to what element formats can be used in its construction, the process itself should be refined until it is as predictable as possible.

Conceptual standards

Conceptual standards can be much trickier since this deals more with subjective issues than objective ones. Conceptual standards typically focus on the type of information to be included as well as the look and feel of the final product.

It is rare for conceptual standards to be established without many discussions among the author community.

Difficult to access information

By far, content experts must be relied upon to provide accurate information regarding their projects, but SMEs may only be available in remote locations or for limited periods of time. It is essential for the success of any enterprise-wide system that SMEs or their their counterparts be made accessible for data update and renovation.

Inconsistent document structure

In the past, this has been addressed to a greater or less degree in word processing templates, but templates are not constructed with rules for their use. Templates can be easily maodified to suit an exemption instead of left inviolate to create consistent documents.

Further, templates are designed to address the issue of appearance but are useless as regards structure and order of the contents.

Limited information interchange

Although copy-and-paste functions provide a way to share information across multiple documents, traditional methods of storage do not provide a means, secure or otherwise, for searching the contents of documents and retrieving these elements for insertion in multiple information products.

Legacy Documents

There are hundreds, perhaps thousands, of documents that comprise the information inventory and intellectual property assets of any corporation. Invariably, not all of these documents support products and services included in current offerings and, as a result, it is imperative to isolate only the most valuable documents for conversion and update to XML methodologies.

A common method of separating documents into various categories of importance is to review the content relative to the revenue generated by the product it supports. Often, this leads to three, or at the most four, classifications of legacy documents.

Type 1

These documents support the currently prodcuts that represent the bulk of a company's revenue. These are the information products that will be converted to XML.

Type 2

These documents support mature product and services that are still actively used by the client base but are beginning to readch the end of their life cycle.

These information products are typically stored in the their original form, such as MS Word format. Documents for which source code cannot be located are usually scanned to OCR, thereby recreating editable text.

Type 3

These documents support products that may still be in use but, collectively, represent old product due for replacement or discontinuance. These documents are most often scanned to .PDF, or some other equally inviolate format, and made available to clients on an 'as needed' basis.

Type 4

These are documents for obsolete products. Almost without exception, these are scanned to PDF format and archived

The Challenges

Although the following list is by no means complete, these issues represent the primary challenges facing documention professionals in all industries:

Faster production and delivery

This assumes that authors are well-trained and competent with the tools. Given the fact that XML authoring is relatively new and the methodologies are still in the development stage, accelerated production may not be a reality in the near term. Ultimately, however, XML tools and process offer the best chance for maintaining an 'evergreen' document perspective.

Increased information accuracy

Although SMEs will undoubtedly continue to supply content for the foreseeable future, the concept of modular documentation has the potential to preserve the veracity of existing information when it is concantenated with new data in the future.

Use of multi-media

More than any other single aspect of documentation, multi-media offers the possibility that informatin presentation will be tailored not only to the needs of the user but also to the essence of the content. With options such as animation, film clips, audio and interactive browser-based testing, the form of the information can be more closely matcxhed to the function of the content.

Development of best practices and standards

Of all the challenges, the lack of best practices and standards poses the greatest threat to timely document delivery and suggests the greatest risks if it not implemented. Especially for an enterprise that is scattered across many product centers and several countries, standardization is the key to raising the level of documentation effectiveness while, simulaneously, delivering more accurate and timely information.

Timely response to end-user needs

Standardization is fundamental this issue, but in order to deliver information to the user on a timely basis without regard for the specifics of which product or who authors the document, a delivery mechanism is needed that can quickly and inexpensively get the information to the user.

A centralized document repository and distribution system should be able to deliver whatever form is requested, including hardcopy, CDs and web-based presentations. An effective and efficient repository provides many critical functions:

- Archive of obsolete information
- Search and retrieval of current information
- Harcopy with a common look and feel, i.e.; format, size, binder type, cover pages, disclaimers, graphics, etc.
- CDs with automated installation scripts and relaiable operation
- Browser-based interactive presentations that include automated feedback of the results to supervisors.

Reusability of data

At the heart of a modular XML approach to documentation is data reuse. If this cannot be accomplished, the greatest benefit of an XML modular approach is for naught. Data reusability is the fuel that drives reduced document production time, increased content accuracy and timely information delivery to the user.

Significant pre-production planning

Traditional information products have been created by copying and pasting data from one document to another in addition to writing new content. The information typically flows into a template that manages the form and appearance of the document but not the structure. But the concepts of modularity and document deconstruction demand both forethought and skill prior to the creation of a document.

Traditional authoring practioces commonly consume about two-thirds of the production schedule for gathering data and authoring the document, leaving one-third for the review, approval and publishing aspects of the process.

Information products built from modular reusable components, on the other hand, demand that 20% of the schedule be devoted to data acquisition, 40% to document deconstruction and authoring, 20% for approval and review, and 20% for publishing and distribution.

This major change in approach to the creation of information products has little chance of success without a clear understanding of the goals and mechanics of the process. Document deconstruction and ‘evergreen’ information are realities that depend almost entirely upon pre-production activities.

Acquiring document deconstruction skills

This can only come from training and a thorough understanding of how information is reused and updated.

Document Creation

Information Production Types

Historically, there have been three basic philosophies guiding the creation of documentation – serial production, parallel production and branching, or optimized, production. Often, the choice of which method to use has been dependent upon the industry and its product development needs.

Each offers unique advantages.

Serial production

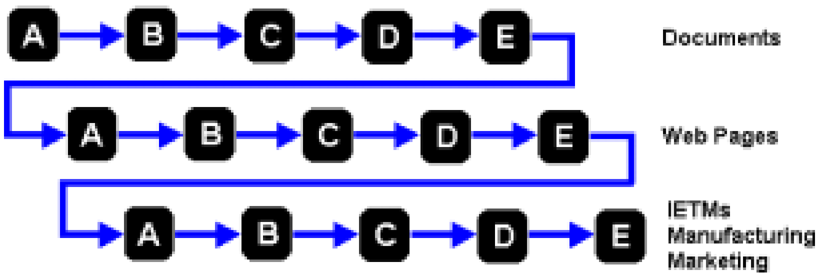
Many technical manufacturing firms, for example, employ the serial method due to its heirarchy of documentation work flow. For these companies, the serial method takes advantage of the increased number of checks and balances to help insure the accuracy of the data and produce the most fundamental documents first.

Drawbacks

- Each product depends upon the internal delivery of the previous product
- Maximum duplication of effort
- Longest time of production
- Most difficult and time-consuming to update
- Highest cost

Advantages

- More reviews of information accuracy.
- Author community is larger and more diverse



Parallel production

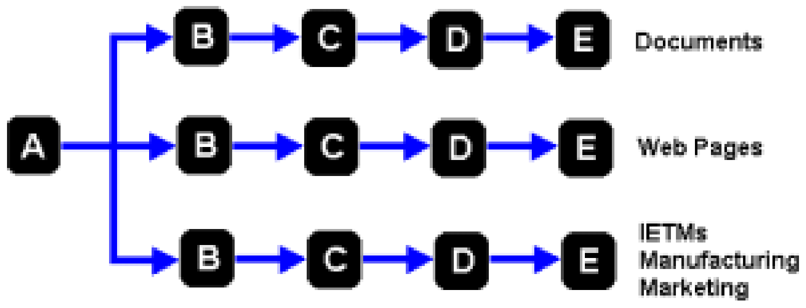
The parallel process model lends itself to companies that simultaneously develop mutilple products that will be used both independently and as part of a suite of integrated products. This method can often reduce the total time to market of related products and their associated documentation.

Drawbacks

- High cost
- Duplicated effort reduced, but still significant

Advantages

- Synchronized design
- Cross-functional problem solving
- Reduced redundancy



Branching production

Branching, or optimized, document production is used to its best advantage in a project-oriented environment, such as software development. Often isolated from other development timelines, a project-oreiented environment brings together many resources for a specifc purpose and for a defined legth of time. Within this environment it is imperative that all tasks, to the degree possible, proceed simultaneously towards a known deadline.

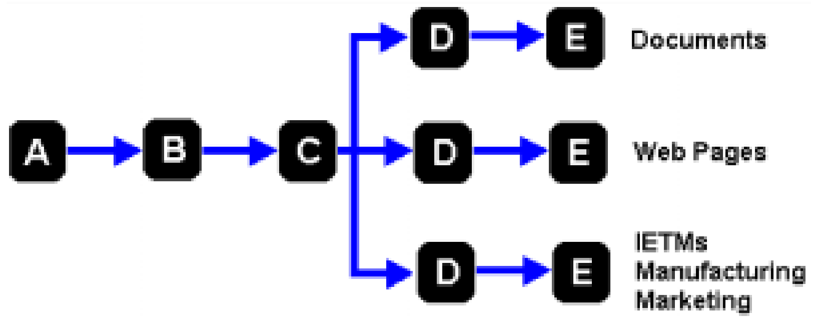
The branching method provides a platform for all documentation to proceed at a rapid concurrent pace during the second phase of a project timeline.

Drawbacks

- Significant commitment from SMEs in the early stages
- Pressure of deadlines in the latter stages of product development

Advantages

- Cross-functional problem solving
- Branching to product-specific products
- Lowest degree of redundancy
- Minimal effort and duplication cost



Branching production dependencies

- Maximum flexibility during product development
- Standardized syntax and terminology must be in place
- Tained authors and fully functional tools
- Established and appropriate documentation delivery systems

Gains from using a branching production strategy

- Quantitative:
  - Reduced effort of approximately one-third
  - Reduced document inventory
  - Reduced non-substantive information
  - Less complex method of updating data
  - Faster republishing and delivery of updated documents
  - Simultaneous updates and data enhancements to multiple documents
- Qualitative:
  - Greater significance of included information
  - Increased levels of understanding
  - Feedback to primary knowledge provider

Modular Documentation

Redefining concepts is critical

When considering a conversion from traditional documentation methodology to modular XML authoring, it is important to recognize the need to redefine log-standing documetation concepts.

A change in the basic perceptions of how information products are created will result in a smoother transition to more complex methodologies.

The idea of computer-aided data entry must be replaced with the idea of **Knowledge Processing**. In modular XML documents, the author must own the concept of dealing with knowledge modules.

Documents and courses must be thought of as **Information Products**. With modular XML authoring, it is possible to distribute information in multiple formats from a single source dicument.

Continuous Production as opposed to batch creation means that information products can be kept current on an on-going basis instead of waiting until sufficient data has been accumulated to make a revision worthwhile the effort.

Parallel and sequential production becomes **Concurrent, then Branched (Optimized) Production** in order to minimize both the cost and time-to-market delivery.

Business case concerns

Finally, there are several issues related to the financial aspects of implementing an XML strategy. Among them, the most significant concerns are:

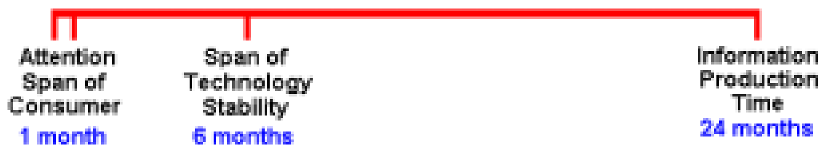
- The costs of tools and resources
- Modular information concept training
- DTD development
- Modular information tools training
- Implementation ramp-up time
- Classification decisions by use, i.e.; engineering, marketing, manufacturing, etc.

Delivery and Distribution

Tools and methodologies provide a way for authors to create new documents as well as regularly update existing information, but a reliable publishing method is vital in order to deliver the information to the user on a timely basis.

The Past

There has always been a huge disparity between what the user wants, what technology can provide and the production time necessary to deliver quality information products. In recent years, the advancement of technology has become less and less of a limiting factor, but document production time continues to lag far behind the desires and needs of the user.


Attention Span of Consumer
1 month
Span of Technology Stability
6 months
Information Production Time
24 months

Realities

There are many factors that directly affect the production of information products. Each of these factors present challenges to the author and benefits to the user but, due to the difficulty inherent in trying to address so msny issues at once, production time remains a stretched-out, sometimes stressed-out process.

- Feature-hungry users who have glimpsed what technology has the potential to provide
- Feature-poor information products that could be signifincantly better if authors and document managers better understood how to use existing and emerging tools
- A constantly changing web infrastructure that includes protocols and browsers
- An increasing bandwidth that allows for larger and more fully-featured documents. When broadly distributed, the information products present download and viewing issues for users with less than cutting edge hardware and those in remote areas.

- Software systems that are typically replaceed twice annually results in constantly upgraded capabilities while presenting the need for a continuing learning curve.
- A wider variety of display devices complicates publishing and distribution issues
- The most difficult problem is “NOW” delivery. When all other issues and considerations can be addressed with tools and training, this single problrem remains as the biggest challenge and the singular goal of all the production efforts.

Implementation

As mentioned before, the obstacles to implementing an integrated, fully-functional authoring and publishing environment are numerous and significant, but each one can be resolved with sufficient planning and the establishing of standards.

Of all the obstacles - both perceived and real - to an effective XML implementation, the greatest challenge is to make sure that effective standards are developed established prior to tackling the mechanics of tool selection and layering methodology.

Implementation issues

The issues surrounding imnplementation of an XML authoring environment can be divided into three broad categories – standards, content integrity and monitoring.

Develop a plan

Although planning will be discussed in detail in the section about project management, a word or two here is in order. You must create a plan! Not only that, but it must be realistic and, most importantly, it must be followed unless compelling reasons to the contrary arise.

This may seem to be a basic tenet but planning is something that most people do only under duress. But a solid, workable plan is essential to achieving a successful XML implementation. There are simply too many details and the time frmae can be quite protracted from initiation to completion.

Standards

Define the current state.

It is impossible to know how to get to your detination without first knowing your origination point. Take a detailed inventory of existing documentation and the processes currently in place that are used to produce it. This exercise must be quite detailed since the goal is to completely replace the existing system. It is critical to identify and understand current procedures in order to determine several points :

- Are there established processes?
- If structured production processes exist, are they being followed?
- What holes, if any, exist in the current system?

Define the desired new state

The creation of standards involves many considerations that are driven by the resources and budget available, time constraints and the desired output.

Who will define the desired standards? This should include as diverse a cross-section as possible of documentation management, authors and selected vendors. Although decisions by committee often have mixed results, it is imperative to gain input from as many perspectives as possible. The discussion should include a mix of representatives from various departments or functional areas since each segment will have unique documentatin needs.

Who will work to develop the standards? Traditionally, documentation managers have been the ones to establish the standards for document structure and templates but, in the XML arena, it is wise to include authors in the discussion after some of the basics have been establsihed.

Develop an approval process that includes a timely feedback mechanism. A well designed system includes the procedures and measures by which documents are promoted and demoted during the creation and editing cycles. Errors cost approximately 3x to fix after delivery.

Anticipate obstacles and develop solutions by contrasting the existing system’s definitions with those of the desired new state. This may emphasize the need for more plannig, or it may verify the veracity of implementation decisions.

Content integrity

Protecting the substantive and modifying non-substantive information is one of the fundamental concerns addressed by modular documentation methodologies.

SMEs must be intimately involved in their areas of expertise in order to accurately ferrret out information that msut be included from that which should not. Without a high level of

SME involvement, one of the most significant advantages of modular document creation is lost.

In addition, the various components of the system – tools, DTDs, FOSIs, standards – must be tested to be certain that included content is presented in a way that helps to generate the greatest degree of user comprehension.

Monitoring

Measures and metrics

Like any project, milestones must be established in order to know to what degree goals are being achieved. This topic is addressed in detail in the section on project management, but a few critical issues cannot be ignored :

Measures should be both qualitative as well as quantitative. Even if a predictable implementation is achieved, it has little significance to the user if the end result has changed very little. Both the visual aspects of the information and the ease with which it can be accessed are important and must be developed in tandem.

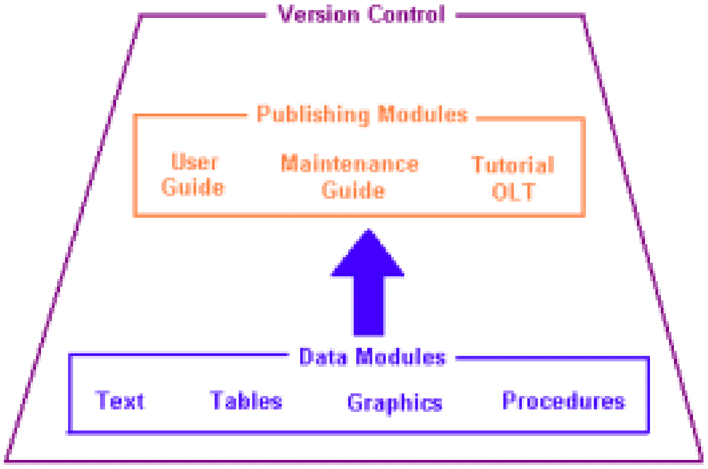
Tracking activities

An effective method of tracking progress will include an effortless input method for authors and management. An existing help desk system may work well in some cases but, often, a separate tracking method is needed until the implementation has reached a point where it is predictable and reasonably trouble-free.

Structured Content Creation

The Framework

The three primary elements of modular XML authoring are data modules, publishing modules and version control. When viewed as a pyramid, it is easy to understand how one element lays the foundation for the next.



Data modules

Data modules are a combination of text, tables, procedures, static graphics, animated graphics, movie clips and audio. Typically, a data module is limited to one complete thought that includes several elements.

For example, a typical data module might contain a title, several lines of text, a step list, several graphics and a few more lines of text.

Although a data module is sometimes compared to a content unit in theory, the reality is that a data module is almost always the result of a group of content uints.

Publishing modules

A publishing module is the framework that organizes data modules and to which an output FOSI is mapped. It is the publishing module that, when completed, results in a ‘finished document’ containing all of the information needed for distribution.

A publishing module not only links together data modules in an order that supports the information being authored, it also provides a platform for linking information contained within one data module to information contained within another module. This internal linking capability of a publishing module provides two significant benefits:

1. Data modules can be cross-linked in the same way that hyperlinks are used in traditional authoring tools
2. The information contained within data modules linked to a publishing module can be edited or replaced, resulting in a fully updated document the next time it is published.

Version control

Version control is the final key to the creation and maintenance of an ‘evergreen’ documentation system. It is through the mechanism of version control that only the most current version of a data module is made available for linking to a publishing module. As a result, when a document is released for distribution, only the most current version of the publishing module is used by the output FOSI.

At some predetermined point a document can be ‘locked’, creating a baseline for all future iterations of the document. Until it is replaced, the locked version of a document is the version that will always be mapped to an output FOSI if additional published product is needed.

Benefits

Structured content creation offers many enterprise-wide benefits that derive directly from standardization, including:

Replication

A unified structure and creation process that can be taught to new authors as documentation communities grow or resources are replaced.

Reuse

Information reuse, of course, is at the heart of modular authoring. Although as yet not quantified, the potential for substantially reduced document creation time is obvious as the amount of information that can be reused increases. Even given initial delays in document production due to expected learning curves and implementation issues, informatino reuse offers the greatest return on the effort invested in documentatin and support.

User-focused output

Simplified multi-format publication, also known as single-sourcing, can only be provided by a system that can link different output templates to a document. The concept of FOSIs was developed to answer this need and, at the same time, provide a consistent appearance to distributed documents.

Creating a variety of output FOSIs that focus on specific applications of a document allow changes to be made to one FOSI without affecting either the document or any other FOSI.

Single-sourcing also eleiminates the delays customarily encountered when editing a document in preparation for republishing to meet new user demands.

Consistency

A standardized look and feel to all documentation has long been the goal of marketing. Maintaining a easily recognized identity in both hardcopy and online information is one of the key elements in projecting a coprorate image that is intentional and consistent.

By removing from authors the subjective decisions regarding document appearance, structured content creation canbe distributed only by means of integrated FOSIs that produce documents with a predetermined look and feel that guarantees

Revising information

It is far easier to keep information “green” when data can be updated in small chanks. Traditionally, authors wait until changes have accumulated enough to make the revisin process worthwhile. Unfortunately, this delay often results in data that is no longer valid by the time it reaches users. In a worst-case scenario, the data will require yet another revision even as it is being delivered with the previous update.

Revising information on an on-going basis assures users will have only the most current data, technology and safety information.

Downside considerations

As might be expected, process control, streamlined authoring methods and publishing consistency cannot be achieved without cost. Inherent in any effort to standardize document creation and publishing is a loss of personal control on the part of the author. By including representatives from the authoring community in the decision-making process, especially as it concerns DTDs and FOSIs, there is every reason to expect that authors will feel they have contributed in advance to the look and feel of the documents they create.

Still, old habits and methodologies sometimes die hard and it is important to recognize and be preapred to deal with at least four unavoidable issues.

Loss of personal style

As a group, writers are fiercly independent and tend to dig in their heels when pressed to respond to criticism of their style and content delivery. The demands of a structured creation process preclude, to a large degree, the possibility that one document will read significantly different than another.

More than any other aspect of a structured methodology is the resistance by authors to create a document that is indistinguishable from its fellows. No doubt less prevalent in non-fiction writng in general, and technical writing in particular, is the fear that an author’s efforts will not stand out among its peers. Often played down or denied, it would be a mistake of the highest order to ignore this reality, pretend the perspective does not exist or, worst of all, hold the opinion that such a position is foolish or immature.

Rather than try to sweep this issue aside, a far more effective approach is to deal with it up front by including every author possible in testing the validity of multi-vendor tool integration and operation performance prior to the full implementation of any system.

The price to be paid for avoiding this problem can include extended implementation schedules, prolonged conversion timelines and a turnover of resources. The latter cost, turnover of personnel, is perhaps the most potentially damaging of all given the high cost of redruitment and training.

Advance planning

One of the demands of a modular approach to documentation is the absolute need for planning. The deconstruction process, the practical application of this requirement, will almost always require more time than the actual creation of modular information.

In traditional authoring environments, a document often flows logically from one aspect of a product or service to the next. IN such environemts, the information can be allowed to organize itself to a certain extent as the document segues from one topic to another and from the general to the specific.

Advance planning also establishes the inter-relationships of the content relative to cross-references and links to external documents.

Control of the display

Diminished focus on presentation formatting due to the application of FOSIs is another way in which personal style is removed from authors who have grown accustomed to viewing a finished document in a particular way.

In the XML environment, elements such as notes, safety warnings, captions and the like are presented by the rules of a FOSI leaving little or no control for display in the hanbds of the author. Like the other issues of personal style, this is a potential cause of a reluctance on the part of an author to cooperate in a methodology that disallows personal influence.

Learning curve

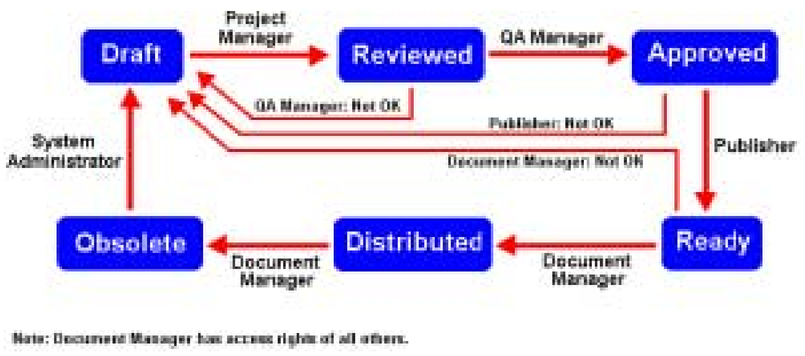
The learning curve of any new product can be anticipated and dealt with providing there is a willingness to recognize the potential problem. With XML authoring tools, most especiall if the implementation includes a fully functional EDMS, the learning curve can be quite protracted.

Only the rarest of circumstances will result in an implementation that is free of integration problems. These challenges, along with new tools and methodologies, can create frustration and delays that can easily exceed the expectation of management. When this occurs, the result is always pushed deadlines and postponed product introductions, both of which directly affect the bottom line of any organization.

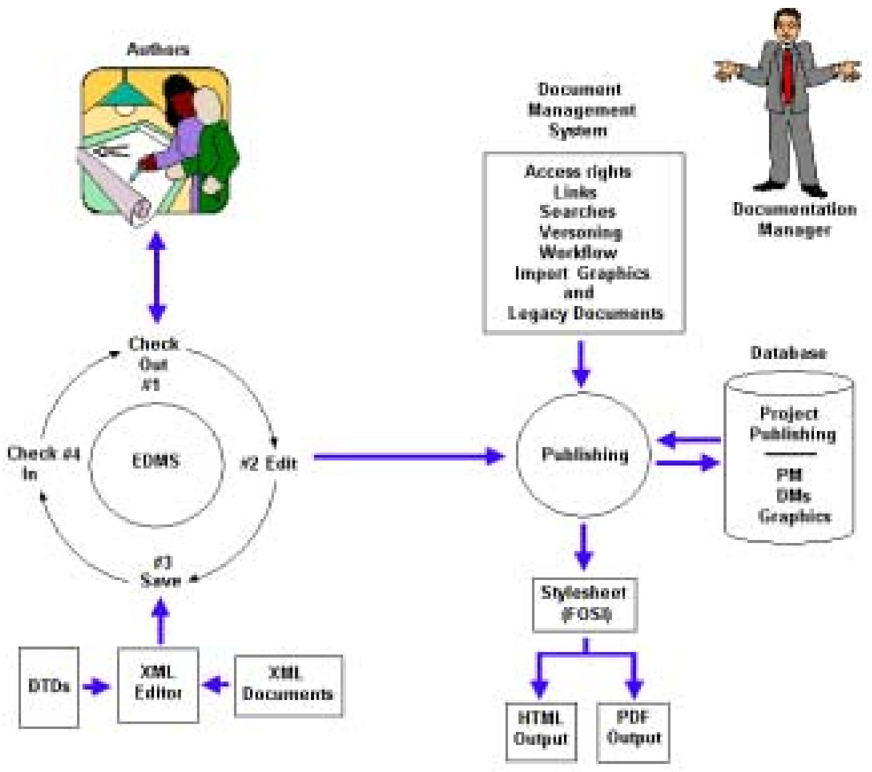
The EDMS Workflow

The Workflow Path

This is an example of the path a document takes as it moves from the author through the approval process and is ultimately published.



The EDMS Process



Information States

The following categories represent generally accepted states for information that passes through a well-defined approval process. In less structured some environments, it may be advantageous to combine two or more states into one.

Draft	This is the state for new information or when editing existing information. Information objects as well as entire documents will remain in this state longer than in any other.
Reviewed	This state focuses on the accuracy of the content. This review may require significant rewrites if the original data was unclear, resulting in a demotion to the Draft state. If this occurs, the information must pass through the Reviewed state again before being promoted to the Approved state.
Approved	In this state, the information is checked for grammar, spelling, punctuation and comprehensibility. It is not uncommon for a document to be demoted to the Reviewed state if major changes to sentence structure have resulted in questionable content accuracy.
Ready	The review at this stage verifies that the document structure is correct and suitable for the purpose for which the information is being authored. For example, if the information is a training tool, the OLT DTD should be evident in the flow of the material and the data module structure should fit this model.
Distributed	This state identifies a document that has been published in any form, including PDF or HTML format, or in hardcopy.
Obsolete	This state signifies documents that have been archived and are currently not in use. An obsolete document can be retrieved and used as a basis for a new document.

User Groups

User groups define the individuals who have access to the contents of the EDMS and what they are allowed to do within the authoring environment. Generally, users can be divided into several groups based upon access privileges.

Readers	Read-only access.
Authors	Create, Edit and Delete access in the Draft state Read-only access above Draft state
Reviewers	Edit and Delete access in Draft state Read-only access above Draft state
Approvers	Read and Edit access through Reviewed state Read-only access above Reviewed state
Publishers	Read and Edit access through Approved state Read-only access above Approved state
Document Managers	All access rights through Ready state Read-only access above Ready state
Administrators	All access rights through all levels.

Roles and Responsibilities

Readers	Readers are usually either SMEs who determine whether or not the content of an exisitng document should be updated, or Project Managers who must make an assessment about documentation needs.
Authors	Documentation specialists as well as project engineers and developers. Often, SMEs provide the initial content for a project’s documentation.
Reviewers	Project Managers and Product Champions are usually the best resource for validating the accuracy of technical content. Knowlegable authors may also find themselves in this role.
Approvers	The Quality Control Manager for a project may share these durties with with representatives from a corporate Quality Control department

	and the legal department. The primary function of Approvers is to verify that corporate standards have been appropriately met.
Publishers	Document Managers and Documentation Specialists are usually the best resources for validating that documernts have been constructed correctly and, as a result, will output correctly.
Document Managers	These resources are divisional Documentation Managers and lead departmental Documentation Specialists with the ability to distribute and obsolete documents in addition to all other activities. Doc Managers can also create projects and project containers.
Database Admins	Deleting information objects, managing old versions of documents and creating new projects are the primary roles.

Authoring Concepts

Document Deconstruction

Basic considerations

There are two fundamental reasons for implementing an XML strategy - reusability and ease of revision. A document that can be readily updated will be edited and republished more frequently, thereby providing a consistently higher degree of timely information to users. This concept is often referred to as an ‘evergreen’ approach.

Some information is best authored in a module while other information is best suited to inclusion in the base doucment. An integral part of the document creation and deconstruction processes is the determination of what data needs to be placed where for easier access to revisions.

Module or not?

There are four questions that can facilitate decisions regarding what should, and should not, be authored as a module:

- Will the information be reused?
- Will portions of the information be reused?
- Will the information be updated frequently?
- Is it a procedure?

The determination of whether or not a specific piece of information should be authored as a module comes down to this simple rule:

> If the answer is ‘YES’ to any one question, then author it as a unique module.
>
> If the answer is ‘NO’ to all of the questions, author the information in the base document.

Ideally, this question-and-answer technique would be applied to the entire contents of a document prior to deconstructing and authoring any of it.

Information types

Document elements can be further broken down into three primary categories based on the type of information contained within the module – procedures, topics and library items.

- A procedure is a list of steps or a squence of actions that the user must take in order to achieve a specific result or operaton goal.
- A topic is a discussion of one subject and is usually only a component of a section or sub-sectin within the document.
- A library item is a piece of information, such as a table or a graphic, that is created external to the XML authoring environment and then imported and linked to a module.

It is good practice to nest procedures and heading hierarchy no deeper than four levels but, in its most complex form, documents with six levels are not unusual.

Content unit

This is the smallest useful chunk of information on a specific topic, and can include multiple elements such as text, graphic, tables and procedures.

Although, technically, a content unit can be as small as a single sentence it is rarely less than a title, explanatory text and a graphic, table or procedure.

Containers

A receptacle used for a collection of Information Objects, this is often likened to a folder. Unlike a folder, however, each container is meant to hold only one specific type of information.

Containers are the organization method employed within an EDMS to create, segregate and track specific types of authoring units. A container can be any one of the following types, all of which are included in every document project:

- The Project container is the root container and holds all of the other types.
- The PM container holds the underlying document, or documents, for a project. The Publishng Module is a framework into which all data modules are linked.

- The DM container holds topic modules, procedure modules, tutorial modules and lesson modules.
- The GM container holds imported graphics of any 'legal' format, including PNG, SVG, JPG and, evetually, PDF.
- The Legacy container holds imported documents from external sources such, as Microsoft Word or Excel. These documents are stored in their original, or native, format.

Information Object

An Information object is a standalone unit of data that can be a graphic image, a table, a formula or any piece of information that will be an element of a content unit contained within a data module.

An information object resides in a container library within the EDMS and is used by linking it to a topic module. An information object is not usually text, except in instances when the text is reused in multiple modules without modification. Examples of text information objects are security alerts, safety warnings and medical remedies.

Information Modules

Different types of modules are created to hold specific types of information and are authored according to the DTD for that particular information type.

A procedure module is the authoring unit for creating a step list or a set of instructions that must be carried out in a specific sequence. A procedure module will typically contain a title and explanatory text related to the procedure, and may also include graphics or a table. Procedure modules can be nested inside one another or they can be nested within a topic module.

A topic module is the authoring unit of choice when writing about a single subject that does not include procedural or process-related information. Topic modules can be nested inside other topic modules, and can create a hierarchy of up to six levels.

A lesson module is the authoring unit for creating one topic in a series of instructional presentations. A lesson module is rarely used for purposes other than online training or other technology-based training applications.

A tutorial module is the authoring unit used when creating one page of a single topic within a Level 1 or Level 2 tutorial.

Additional functionality

Enhanced functions such as cross-references, external document linking, captioning, list numbering, etc. are engineered into the DTD.

Titles and labels stand apart from a content unit but are included in an information module as part of the topic.

Document Type Definition (DTD)

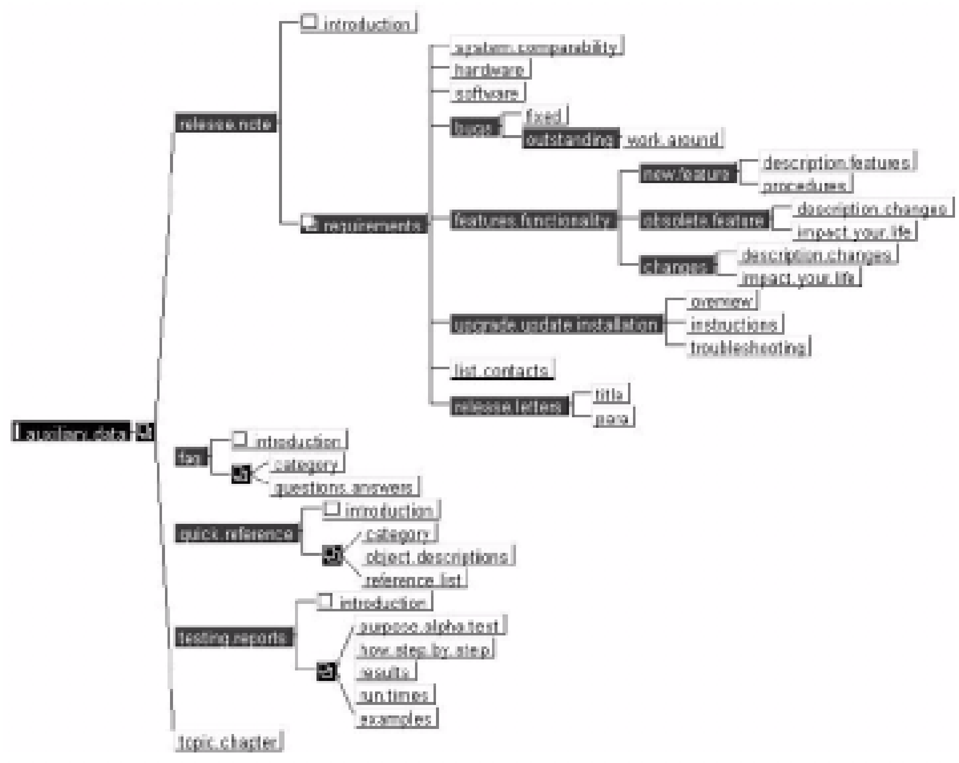
Features and functionality

A DTD defines the structure of a class of documents and provides information about many aspects of the document, including

- Hierarchy of document elements
- Element attributes
- Entities
- Notations and their attributes

A DTD includes rules about what types of content are allowed and how it must be written. A user creates a document instance of the DTD, much like a template is used in Word. But unlike a template, a DTD cannot be modified by the author.

Below is an example of a portion of a software manual DTD. Note that the hierarchy requires that different types of content be placed in specific locations within the chapter or section.



A DTD is not necessary for XML authoring, but it is needed for standardized technical authoring. It’s use results in a well-formed, syntax-correct document. A DTD often includes optional topics that can be included, or not, at the discretion and need of the author. With each topic that is included in the document, howver, come rules for its use and the particulars regarding what material may or may not be included.

One of the greatest advantages of using a DTD is that authorized operations within the documents can be verified at the time of publishing.

The Authoring Environment

A complete authoring environment is comprised of several elements that have been successfully integrated – the editing environment, the publishing environment, DTDs, FOSI, and the electronic document management system.

Tools of the Authoring Environment

Hidden complexity

A complete authoring environment brings together different tools for specific purposes in order to create a complete system.

Word processors such as Microsoft® Word™ and Corel® WordPerfect™, and desktop publishing packages such as Adobe® PageMaker™ and FrameMaker™, are developed and shipped to the user as a complete system with layout, authoring and publishing tools integrated into a single product.

Conversely, the XML authoring environment must be assembled from components that provide specific functionality. These components can be brought together from multiple vendors or they can be acquired from a single source but, in either instance, they must work together seamlessly either by design or customaization.

It is not unusual to find that the friendliest XML editor will not interface with a fully-featured publishing system, or that the best FOSI tool cannot be successfully integrated into a desirable command language processor.

Which system to choose

If an XML authoring tool will be used to create new documents and convert exisitng documents for display in a browser and nothing else, as might be the case for a compnay's intranet, all that may be needed are a single FOSI and minimal command (macro) language capabilities.

But if a more robust system is needed that must include version control and multiple output formats, the selection of the correct tool becomes a more arduous task.

The decision to implement an EDMS is often the determining factor behind the choice of which XML authoring tools to use. Since there are fewer available options for an EDMS than there are for XML editors, and even fewer choices when customization capabilities are considered, it is easy to understand why the management system may be the first consideration when building an XML strategy.

Adept EPIC XML Editing Suite

The Adept Editing Suite is presented as an example of an integrated, full-service system providing a high degree of flexibility in the publishing side as well as in the authoring environment.

Bear in mind that the Epic suite satisfies all of the needs for a complete authoring system but does not address the issue of information management and version control. These functions must be provided by an electronic data management system (EDMS) that can be made to integrate with the authoring tools.

Epic+ Editor

This is the authoring tool for creating XML documents. The editor uses a set of predetermined content tags and a DTD that have been built using one of the other tools in the suite.

Epic Publisher

This sister tool to Epic+ is a composition component for formatting and printing XML documents. It is used in conjunction with an output style sheet that maps tags in the authoring environment to specific publishing displays such as bolding, indents and numbering.

Document Architect

The developer's tool is used to build an XML application for use with Epic+ or Epic Publisher. The application includes rules for DTDs and a style sheet as well as customization routines.

Epic Command Language

ACL is a programming language providing over 400 functions and commands that allow developers to customize the interface or develop complete applications.

Epic Series

***F**ormatting **O**utput **S**pecification **I**nstances* (FOSIs) provide different styles for different display purposes, including local display, hardcopy and web browser.

Since XML data contains no formatting information, style sheets are used to format the data to produce electronic, printed and other output. There can be as many style sheets as needed; i.e.: pocketsize and full-size publications, HTML, PDF, etc.

Building a Sample Project with the Epic Editor

Process Overview

Whether or not the documentation project involves an existing document or a new one, the process is the same.

1. Create headings within the document to no more than four levels
2. Write and edit the content
3. Review the content for grammar, punctuation, spelling and comprehensibility.
4. Create a Table of Contents.
5. Return the draft to an SME for accuracy review.
6. Make the final corrections.
7. Create a document deconstruction map.

Deconstructing the Document

- Segregate blocks of information by content, either by means of the Table of Contents or by reviewing the content of the document.
- Determine the hierarchy of the information by again referring to the Table of Contents or by reviewing the document content.
- Determine whether the information is significant enough to warrant the creation of a data module, or if the material requires only a content-unit that will ultimately be included within a data module.
- If a data module is needed to contain the information, determine which type of module is appropriate – topic module, procedure module or lesson module.

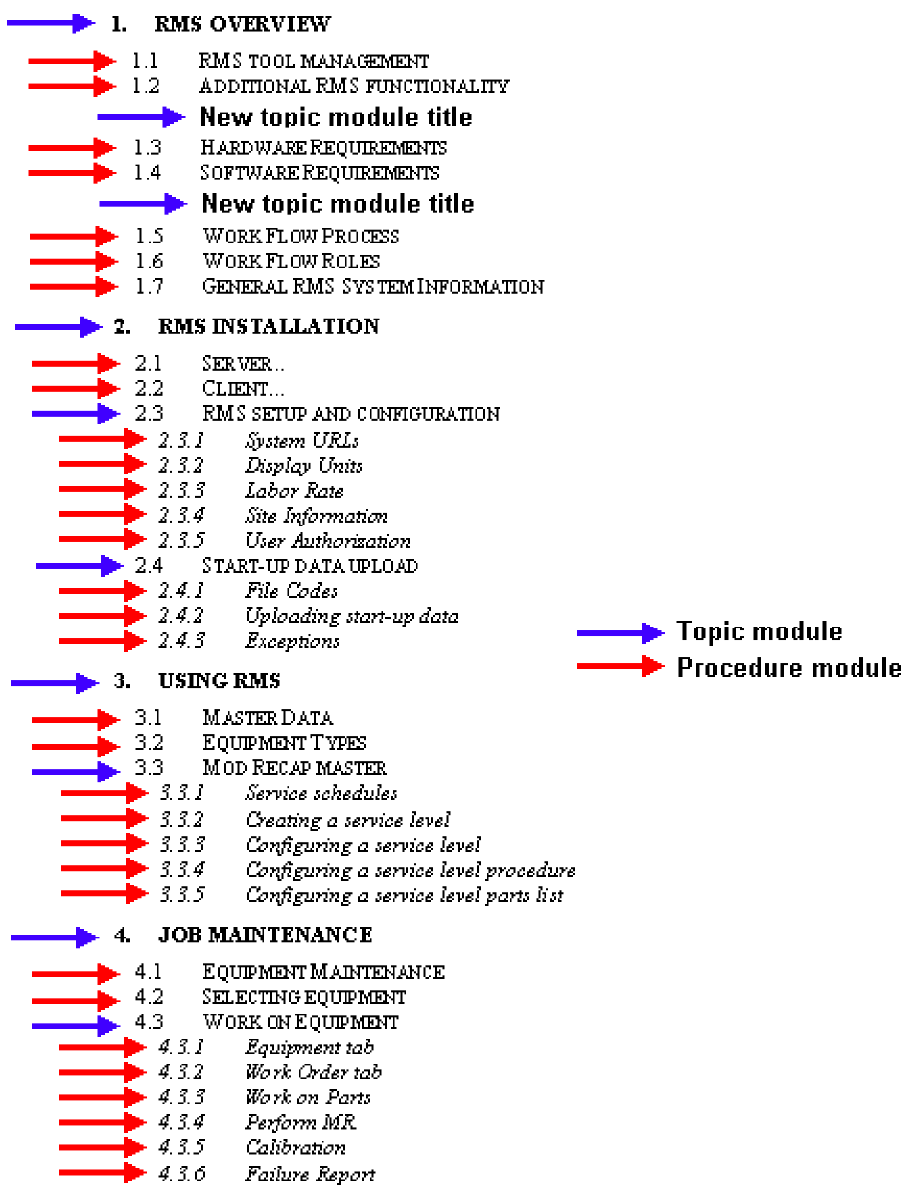
  Typically, if the information is a procedure or a list of actions, the data module of choice is a Procedure module. If the content presents any other type of information, a Topic module is appropriate.

Note: A procedure is authored as a step-list within a procedure module. If topic modules must be nested, they can only be nested within other topic modules.

- Label each block of information with the module type, i.e., Topic, Procedure or Lesson.

A sample deconstruction map is shown below.

Note that the hierarchy of an XML document is similar to, but rarely exactly the same as, the hierarchy of a document authored in MS Word. This is due to the considerations of modular documentation and the structure imposed by the DTD in use.



🟦 **1. RMS OVERVIEW**
- 🟥 1.1 RMS TOOL MANAGEMENT
- 🟥 1.2 ADDITIONAL RMS FUNCTIONALITY
- 🟦 **New topic module title**
- 🟥 1.3 HARDWARE REQUIREMENTS
- 🟥 1.4 SOFTWARE REQUIREMENTS
- 🟦 **New topic module title**
- 🟥 1.5 WORK FLOW PROCESS
- 🟥 1.6 WORK FLOW ROLES
- 🟥 1.7 GENERAL RMS SYSTEM INFORMATION

🟦 **2. RMS INSTALLATION**
- 🟥 2.1 SERVER..
- 🟥 2.2 CLIENT...
- 🟦 2.3 RMS SETUP AND CONFIGURATION
  - 🟥 *2.3.1 System URLs*
  - 🟥 *2.3.2 Display Units*
  - 🟥 *2.3.3 Labor Rate*
  - 🟥 *2.3.4 Site Information*
  - 🟥 *2.3.5 User Authorization*
- 🟦 2.4 START-UP DATA UPLOAD
  - 🟥 *2.4.1 File Codes*
  - 🟥 *2.4.2 Uploading start-up data*
  - 🟥 *2.4.3 Exceptions*

🟦 **3. USING RMS**
- 🟥 3.1 MASTER DATA
- 🟥 3.2 EQUIPMENT TYPES
- 🟦 3.3 MOD RECAP MASTER
  - 🟥 *3.3.1 Service schedules*
  - 🟥 *3.3.2 Creating a service level*
  - 🟥 *3.3.3 Configuring a service level*
  - 🟥 *3.3.4 Configuring a service level procedure*
  - 🟥 *3.3.5 Configuring a service level parts list*

🟦 **4. JOB MAINTENANCE**
- 🟥 4.1 EQUIPMENT MAINTENANCE
- 🟥 4.2 SELECTING EQUIPMENT
- 🟦 4.3 WORK ON EQUIPMENT
  - 🟥 *4.3.1 Equipment tab*
  - 🟥 *4.3.2 Work Order tab*
  - 🟥 *4.3.3 Work on Parts*
  - 🟥 *4.3.4 Perform MR*
  - 🟥 *4.3.5 Calibration*
  - 🟥 *4.3.6 Failure Report*

🟦 **Topic module**
🟥 **Procedure module**

Reconstructing the Document

A word of caution

It is critical to recognize that the numbering system and order of a Table or Contents created in Word is not a roadmap for an XML document. In point of fact, there is every reason to believe that the various sections of a Word document will ultimately be rearranged to properly create an XML document containing the same material.

This reality is due to the fact that an XML document is constructed by means of a DTD with a specific set of built-in rules and structure. By contrast, a Word template has no provision for disallowing content insertion based upon a predetermined rules set.

Rebuilding a document

Although a legacy document must be deconstructed from the top to the bottom, it is rebuilt from the bottom to the top. Essentially, the procedure is as follows:

1. Author individual modules (topic modules, procedure modules, etc)
2. Link modules to other modules as called for by the deconstruction map
3. Author the appropriate topics directly in the Publishing module
4. Link the top-level data modules to the Publishing module in the correct places.

Creating a New Project

Management responsibility

A new project is created by a member of the Admin team or by the Document Manager in the appropriate technical domain. A new technical domain may be needed if the project will support a new product that does not fit within existing domains.

After a project has been created in the SigmaLink EDMS, the Admin or Document Manager will also create the appropriate containers for the project, including:

- The Publishing Module (PM) Container
- The Data Module (DM) container
- The DM sub-container for each document, if needed
- The Graphics container
- The Legacy container

The Authors' responsibility

After the containers have been created within the project, the author can proceed with the following activities:

- Creating the PM information object shells for each document, i.e., User Guide, Installation Guide, Tutorial, etc.
- Creating the information object (data modules) needed for each document. At this point, the information objects are merely shells awaiting information.
- Selecting the appropriate document model (DTD) for each module, i.e., Topic module, Procedure module or Lesson module.
- Selecting the correct DTD for the document.

Importing Files

If an existing Word document is to be converted to XML, complete the following activities at this time:

- Create a new folder for the document named **DOCNAME HTML**.
- Save the document in the new folder as an HTML file
- Convert the extracted graphics from GIF format to PNG. This is accomplished by using a utility such as HypersnapPro. Graphics must be opened and resaved individually.
- Import all of the PNG format graphics into the Graphics container in the EDMS.
- Import the legacy Word document into the Legacy container.

Styles

Titles

A title should always be followed by text.

The title should never contain only an acronym. Instead, type the term or phrase completely and then follow it with its acronym in parentheses.

Syntax

Use the word “Click” for any action requiring a single left-hand mouse button click, such as “click Apply.”

Use the word “Select” when choosing an item(s) from a list.

Menu selections are noted in a mix of upper and lower case with the “greater than” symbol separating the options:
“**Edit>Copy**.”

Emphasis

- Menu options
- All “clicked” items, such as “click **OK**”
- Hypertext, action buttons and radio buttons
- The title of a Tab

Filenames

Filenames and paths are written with these rules:

- Do not exceed three levels in the path.
- The entire path is noted in capital letters:
  “**FOLDER \ FOLDER \ FILENAME**”

Lists

Ordered lists and step-lists are preceded by an Arabic number.

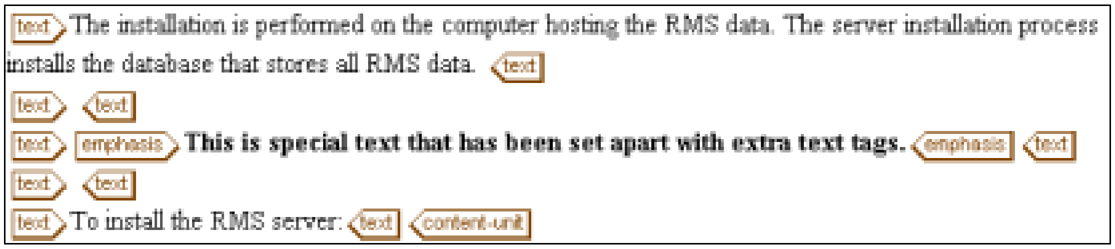
If a step or series of steps is to be repeated, the notation is as follows: “Repeat steps 1 through 7.”

Text Tags

Text tags isolate narrative information and can separate lines of text for purposes of clarity. Text tags are used in three basic ways, the choice stemming from the answer to a question regarding the application:

- Does the block of text present a single idea? If so, use one set of text tags to enclose the entire text block.
- Does the content suggest the need for a simple line break so that a new sentence can begin on the next line? If so, enclose each sentence with its own set of text tags.
- Does a sentence or series of symbols need to be isolated from the rest of the text with blank lines, such as a line of programming code? If so, insert an extra set of text tags before and after the special sentence, in addition to the tags used for the code itself. Place a spacebar space inside the "empty" text tags to prevent an error during the completion check.

  An example of this is shown below:



text The installation is performed on the computer hosting the RMS data. The server installation process installs the database that stores all RMS data. text

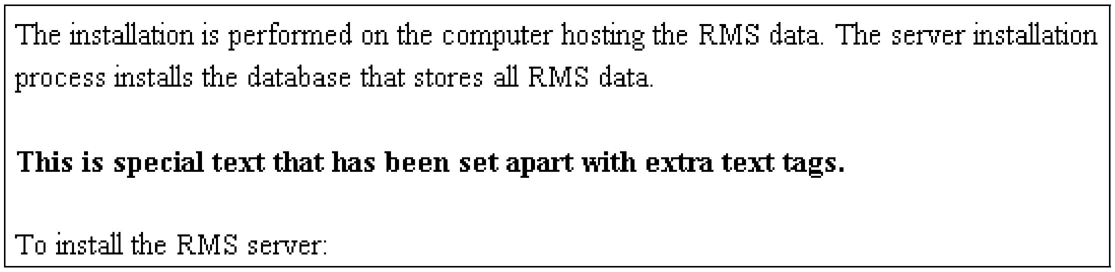
text text

text emphasis **This is special text that has been set apart with extra text tags.** emphasis text

text text

text To install the RMS server: text content-unit

Figure 1: Extra text tags shown in the authoring environment



The installation is performed on the computer hosting the RMS data. The server installation process installs the database that stores all RMS data.

This is special text that has been set apart with extra text tags.

To install the RMS server:

Figure 2: The results of extra text tags in the output FOSI

Graphics

Overview

Graphics must be scaled after they have been linked to an information module. Although it is expected that the process will become more automated in the future, for the near term the process involves several steps.

A graphics editor, such *as Paint Shop Pro, PhotoShop, Illustrator* or *Freehand*, or a screen capture utility such as *Snagit* or *Hypersnap*, is needed in order to establish the proportions of the image.

Note: ***Some viewers, such as*** Paint Shop Pro***, can display the dimensions of each image in the*** File Open ***window without the need to individually open each file.***

Image size

The preferred delivery output is PDF for documents authored in the XML environment. Images for these documents can be any one of three sizes, each of which is expressed in terms of the width of the graphic. In order of frequency, the three sizes are:

- 4.5” – Use this width for most pictures, drawings and screen captures. This width aligns the image with the span of the text. Use this size for almost everything.
- 6” – This width is for large screen captures and pictures with significant amounts of small detail. This width aligns the image with the margins, and the ends of the headers and footers.
- 2” – This is best for small screen captures and drawings as well as tall, narrow graphics.

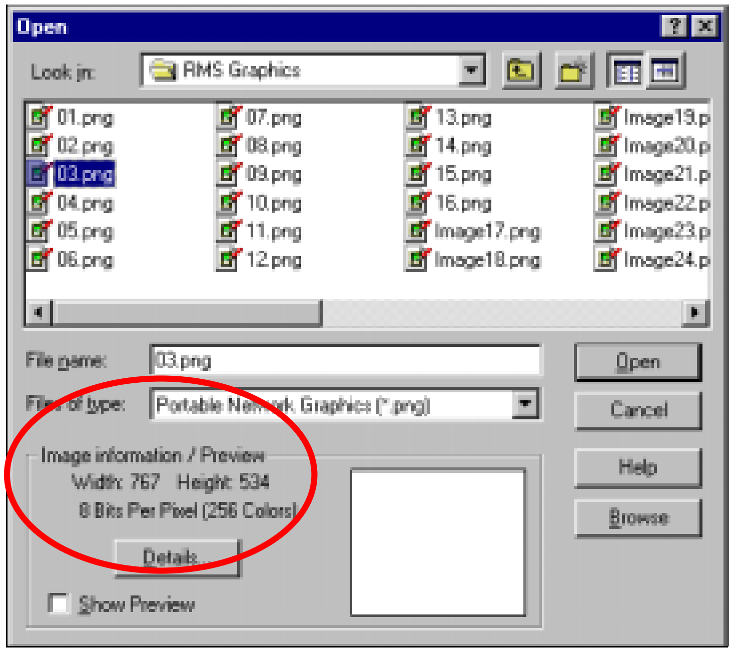
Scaling the image

To scale a graphic within the XML editing environment, it is first necessary to determine a “proportional factor.” The proportional factor represents the relationship of the width to the height, expressed as a three-digit decimal number.

To calculate the proportional factor:

1. Open the image in an external graphics viewer. Paint Shop Pro was chosen for this example and, as a result, the screen capture in step 2 may be unique to this program.
2. Note the dimensions of the image as displayed in the **Open File** window, or print the image and use a ruler to measure the hardcopy version of the graphic.

   In this instance, the dimensions are shown in pixels, but the measurement could just as easily be in millimeters or inches.



File name: 03.png

Files of type: Portable Network Graphics (*.png)

Image information / Preview
Width: 767 Height: 534
8 Bits Per Pixel (256 Colors)

Figure 3: Paint Shop Pro OPEN FILE window

3. Determine a proportional factor by dividing the height by the width. In the example above, the calculation is 534 ÷ 767, resulting in a proportional factor of .696.

Note: ***The proportional factor will always be a number that is less than one.***

As previously mentioned, this number represents the relationship of the width to the height. In this example, the factor means that the height dimension is only about 70% as long as the width dimension.

4. In this example, the height of the image is calculated on a 6-inch width. The height calculation is:

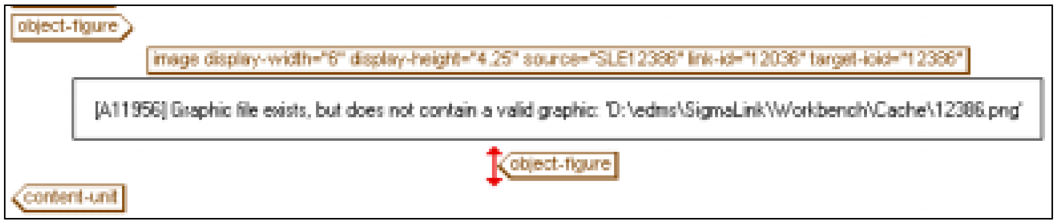
   6 x .696 = 4.176

   or a height of 4.176 inches.

5. Round the height measurement to the nearest quarter-inch. In the above example, the height dimension becomes 4.25.

To enter the dimensions for the graphic:

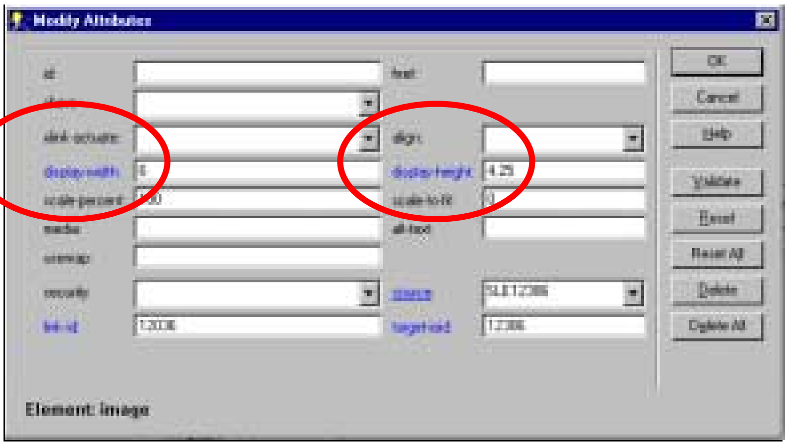
5. Go to the SigmaLink EDMS and check out the module that contains, or will contain, the linked graphic.
6. Insert the link to the graphic if the link does not currently exist.
7. Right-click inside the *object-figure* tags, as shown below.



```
image display-width="6" display-height="4.25" source="SLE12386" link-id="12036" target-ioid="12386"
[A11956] Graphic file exists, but does not contain a valid graphic: 'D:\edms\SigmaLink\Workbench\Cache\12386.png'
```

Figure 4: Object-figure tags in the Data Module

8. When the menu appears, select **Modify Attributes**.



id:		href:	
show:			
xlink:actuate:		align:	
display-width:	6	display-height:	4.25
scale-percent:	100	scale-to-fit:	0
media:		alt-text:	
usemap:			
security:		source:	SLE12386
link-id:	12036	target-ioid:	12386

Element: image

Figure 5: Modify Attributes window

9. Type the width - in this case 6 - as a single digit in the *display-width:* box.
10. Type the height - 4.25 in this instance - in the *display-height:* box using no more than two decimal places.
11. Click **OK** to close the window.

Scaling calculation exceptions

In a few instances, it may be necessary to calculate the width of an image based upon a specified height instead of the other way around. This need will exist when the image has the potential to exceed the page size. In these cases, the proportional factor is calculated in a fashion similar to the norm, but it is applied to the height instead of the width.

For example, the image shown below would probably exceed the vertical page size if the width were set to 4.5 inches, and almost certainly would with the width set to 6 inches.

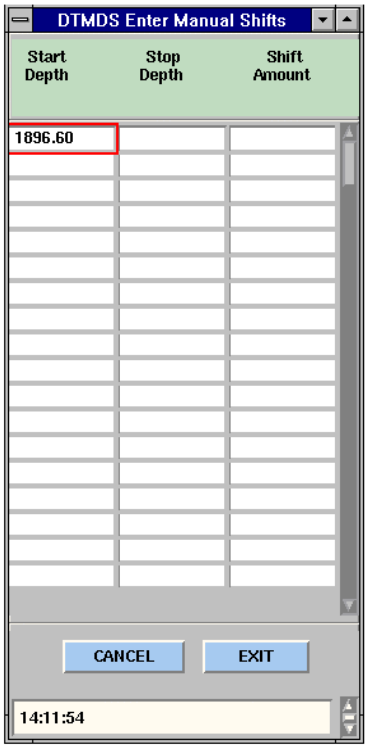


Figure 6: Tall and narrow graphic

To make sure that the graphic does not exceed the vertical page size, use the following procedure:

6. Open the image in an external graphics viewer.
7. Note the dimensions of the image or print the image and use a ruler to measure the hardcopy version of the graphic. In this case, the image is 350 pixels tall and 171 pixels wide.
8. Determine a proportional factor by dividing the width by the height, as before. In this example, the calculation is 171 ÷ 350, resulting in a proportional factor of .488.

   Since the image is taller than it is wide, the factor is applied to the height instead of the width.

9. If the width is calculated on a 6-inch height, width calculation is:

   6 x 2.046 = 2.93

   or a width of 2.93 inches.
10. Rounding the measurement to the nearest quarter-inch produces a final width dimension of 3 inches.

After completing the scaling calculation, link the graphic and enter the scaling data in the *Modify Attributes* window as before.

Editing graphics

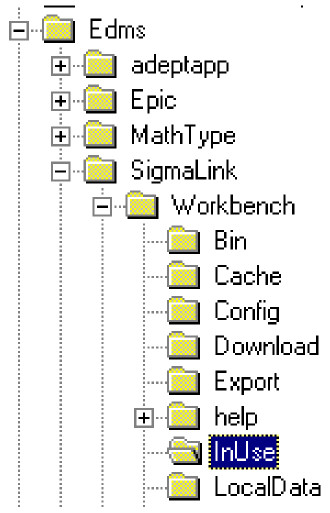
Just like a data module, a graphic must be checked out and then checked back in after is has been modified. This process for graphics, however, has a few more steps in between than a data module.

A graphic, of course, must imported into SigmaLink before it can be checked out modified. Once the image is in the EDMS, it can be modified as follows:

1. Highlight the graphic and right-click to display the popup menu.
2. Select **Properties>Attributes** and make a note of the ID#.
3. Checkout the graphic in the usual way.

Note: *Multiple graphic files can be simultaneously checked out.*

4. Open the file with your graphic editor by navigating to the folder named **\EDMS\SigmaLink\Workbench\InUse**.



5. Retrieve the file with the same number as the metadata ID#.
6. Resave the file to the InUse folder after it has been modified.

***Note**: Be certain to save the file in the same format as the original*

7. Return to SigmaLink and check in the file in the ususal way.

Tables

A table in Microsoft Word is brought into SigmaLink by way of a data module. Unlike graphics, which are imported into the EDMS and then *linked* to a data module, tables are inserted directly *into* a data module.

The following procedure assumes that the data module in question is currently checked out and open in the Epic editor.

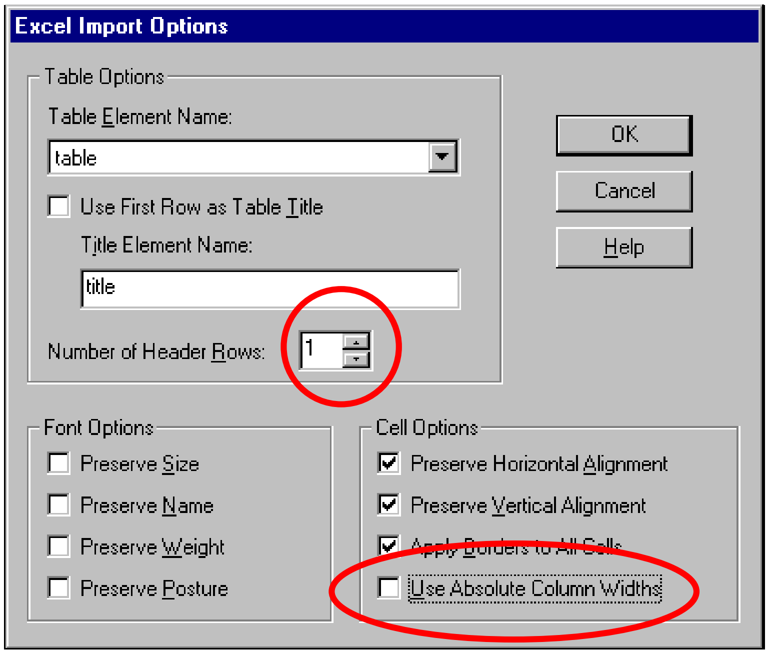
To import a Word table into a data module:

1. Insert a new <content-unit>.
2. Remove the <Text> tags that are included by default with the <content-unit> tags.
3. Convert Word table to Excel spreadsheet by performing the following:
   a. Start Microsoft Word
   b. Retrieve the document containing the table.
   c. Place the cursor anywhere inside the table.
   d. On the Main menu, choose **Table>Select Table**.
   e. Copy the table by pressing **CTRL/C** or choosing **Edit>>Copy** on the Main menu.
   f. Start Microsoft Excel.
   g. Press **CTRL/V** or choose **Edit>>Paste** on the Main menu.
4. In the Excel spreadsheet, adjust the column width of each column to accommodate the full width of its contents.
5. Highlight the cells to be imported into a data module, and only those cells.
6. Press **CTRL/C** to copy the cells or choose E**dit>>Copy** on the Main menu.

***Note**: Font attributes have no affect on the appearance of a published document. Attributes such as size, bold and italic will be displayed in the authoring environment only.*

7. Return to the open data module.

8. Place the cursor at the location of the table insertion.
9. Choose **Edit>>Paste Excel** on the Main menu
10. When the *Excel Import Options* window appears, change only two of the default settings, as shown below:
    a. Increase the Number of Header Rows to "1."
    b. Deselect Use Absolute Column Width



Excel Import Options

Table Options

Table Element Name: table

[ ] Use First Row as Table Title

Title Element Name: title

Number of Header Rows: 1

Font Options

[ ] Preserve Size

[ ] Preserve Name

[ ] Preserve Weight

[ ] Preserve Posture

Cell Options

[✓] Preserve Horizontal Alignment

[✓] Preserve Vertical Alignment

[✓] Apply Borders to All Cells

[ ] Use Absolute Column Widths

11. Click **OK**.
12. Select the entire header row of the table
13. On the Main menu, choose **Table>>Convert to header row**.
14. Click the Check Completeness icon.

Considerations for TBT

Technology-based training (TBT) is typically delivered via either a web browser or on a CD. Documents scheduled for use as TBT are deconstructed in much the same way as any document targeted for XML authoring, but with one exception. These documents must considered from the perspective of educating the user in the operation or maintenance of a tool or application without the presence of an instructor or manager.

To achieve this, the methodologies of instructional design must be applied to the deconstructed content. It is often the case that TBT materials must be capable of providing 100% of the information required to perform a task correctly or prevent injuries on the job.

Like any other document prepared for modular XML authoring, a TBT document is deconstructed to a modular granularity with each module considered for potential reuse. For TBT however, special consideration must be given to the manner of presentation in order to maximize the impact and effectiveness of the training.

Instructional Design Principles

Needs assessment

The first phase of the TBT creation process is intended to define the scope of what is to be accomplished, how much it will cost and the business benefits that will result.

Analysis

This phase takes a close look at the audience demographics, the desired performance goals, the concepts that will be taught, the skills that will be acquired by the user and the types of activities that must be performed.

Design

The focus here is on the training methods. Included in this phase are the development of training outline and the training structure, the transfer strategy after the completion of the training and the plan to get it all done.

Materials development

Prototypes are created at this time which, after validation, provides the platform for developing the actual presentations, lessons and interactions. The materials are tested and final renovations are made in preparation for implementation.

Implementation

This phase begins with a pilot program as well as the creation of train-the-trainer materials, and the beginnings of evaluation and feedback.

Evaluation

This is an on-going process that provides a factual basis for modifications and upgrades to the training system.

Media Decisions

Similar to the questions asked to determine modularity, these queries provide assistance in selecting the most likely presentation media.

- Does it maximize comprehension?
- Does it reduce training time?
- Does it improve retention?
- Does it allow sufficient time for delivery?

If the answer to **any single** question is “Yes,” then the answer is YES.

If the answer to **all** questions is “No,” then the answer is NO.

Project Management

Components of a project

- Resources – people, software tools, equipment and facilities
- Tasks, sub-tasks and duration times
- Schedule – start, task completion dates, milestones and deadline
- Costs and budget – estimated and actual

Considerations when selecting a tool

- Size of the project
- Number of projects to be managed
- Who will access the information
- How will project information be accessed
- What is the end use of project data
- Simpler is usually better

Hidden costs and time-consuming issues

- Create a formalized Product Development Cycle (PDC)
- Organize standards committees to address the creation of DTDs, FOSIs, graphical formats and module granularity.
- Create a test plan for the implemetation

Project Management Tools

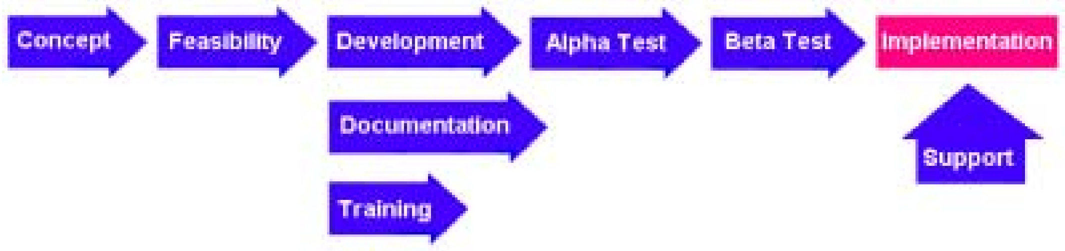
- Microsoft **Excel**

  Although not specifically a project management tool, Excel can provide a simplified means of creating milestones and tracking the progress of small projects or those of short duration.

- Microsoft **Project 98**
  - Off-the-shelf for individual projects
  - Separate file for each project
  - Related projects can be combined as tasks to create a larger project
  - Design a project and track the progress
  - Gantt chart support
  - Administered by a single project manager
  - Inexpensive
  - Relatively easy to learn
  - Common for many clients and vendors
- Pacific Edge **Project Office**
  - Off-the-shelf for multiple projects
  - Handles multiple project managers
  - Integrates with Microsoft Project
  - Very powerful
  - Extended learning curve
  - Expensive
- Microsoft **Access 97**
  - Custom-made for multiple projects
  - Many database applications can be used
  - Can be built to precise needs
  - Extensive learning curve and configuration time
  - Difficult to produce graphical timelines
  - Output is text-based
  - Web interface required for most distributed access
- **OnProject.com**
  - Project management web site for multiple projects
  - Monthly fee schedule based on the number of projects
  - Easily shared with clients and vendors via Internet
  - Data stored in secure external location
  - Support for theaded discussions

The Product Development Cycle (PDC)

The workflow of a typical PDC is shown below. Although ideally initiated concurrent with the Development stage, it is not unusual to for Documentation to begin concurrent with the completion of the Alpha test.



Phase 1: Concept - Defining the project

Purpose

- Describe the perceived problems and challenges
- Clearly and unambiguously state the project and development goal

Deliverables

- Create a Problem and Needs statement
- Target the audience, stakeholders and major players
- List the possible solutions and risks, scope, expected impact
- Prepare a draft of the project plan including budget, resources (templates, standards, tools, web links, bibliography), etc.

Phase 2: Feasibility and Analysis

Purpose

- Collect and analyze information in order to identify user characteristics, such as roles, performance, tasks, etc.
- Gain an understanding of the perceived problems, needs and associated solutions
- Plan the analysis project

Deliverable

- Analysis report

Phase 3: Design

Purpose

- Outline the information and the content structure of the documentation.
- List the training prerequisites, objectives and test items.
- Identify communication strategies including media, data formats and deployment mothods

Deliverables

- A design document
- Working prototype
- Cost estimate and a draft of the proposed budget
- Request for any work that must be outsourced
- Design an operations guide and a user tutorial
- Script and storyboard

Phase 4: Alpha Testing

Purpose

- To determine the extent to which the product design and prototype meet the requirements and goals of the project.
- Plan the Alpha Test project

Deliverables

The Alpha Test validation report includes the following items.

- Level 1 – User reaction and usability testing
- Level 2 – Learning, information processing and usage
- Recommendations and action items

Phase 5 : Development

Purpose

- Execute the design and create a completed product
- Implement recommendations from the Alpha Test report
- Identify resources, risks and constraints
- Finalize the project plan including timelines, budget, roles and responsibilities
- Develop documentation content

Phase 6: Beta Testing

Purpose

- To determine the extent to which the product design and prototype meet the requirements and goals of the project.
- Plan the Beta Test project

Deliverables

The Beta Test validation report includes the following items.

- Level 1 – User reaction and usability testing
- Level 2 – Learning, information processing and usage
- Level 3 – On the job application
- Recommendations and action items

Phase 7: Deployment/Implementation

Purpose

- Distribution of the completed product to users
- Deployment rollout plan, including:
  - Schedule
  - Data formats
  - Contacts and product champions
- Sponsor sign-off on final deliverables
- Post-mortem meeting and lessons learned

Phase 8: Support

Purpose

- Quality of content
- High usability

Support structure

- User support
- Feedback loop
- On-going update and maintenance schedule

Tools and Formats

Function	Product	Format
XML editor/authoring	Arbortext Epic	XML
	Adobe FrameMaker+SGML	XML
	Hypervision WorX SE	XML
	SoftQuad XMetaL	XML
Parser	Balise	
	OmniMark	
	XML Authority	
DTD Development	Near&Far	
Document databases	STEP Electronic Publishing Solutions SigmaLink Workbench EDMS	
	Documentum	
	Poet	
	Oracle iFS	
	Astoria	
Converter to HTML output	HTML-Package	
	DynaWeb	
Composer to PDF output	Adept Publisher	
	Adobe FrameMaker+SGML	
Photo editing	Adobe Photoshop	JPG
	Macromedia Fireworks	JPG
2D vector graphics	Adobe Illustrator	SVG, PNG
	Macromedia Freehand	SVG, PNG
3D vector graphics	ProE Design Automation Option	MPEG
Animation	Macromedia Flash	SWF
Video	Adobe Premier plus After Effects	MPEG1
Audio	Adobe Premier	MPEG3
	Sound Forge	MPEG3
Interactions	Macromedia Flash	SWF
	Macromedia Director	SWF
Testing	QuestionMark Perception	SESSION
Screen captures – video	Lotus ScreenCam	MPEG
Screen captures – static	Hyperion Hypersnap Pro	PNG
	TechSmith Snagit	PNG
	Creative Softworx Capture	PNG
	InBit+s FullShot	PNG
360 degree panaoramas	???	